

Apellidos:

Nombre:

Convocatoria:

DNI:

Examen PED diciembre 2009

Modalidad 0

- Normas:**
- La entrega del test no corre convocatoria.
 - Tiempo para efectuar el test: **22 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		
Para el siguiente fragmento de código C++ de un posible método perteneciente a la conocida clase TCoordenada, la línea "delete b;" liberaría correctamente la memoria dinámica de b. <pre>void Funcion(void) { TCoordenada *a = new TCoordenada; TCoordenada *b = new TCoordenada[5]; (.....) delete b; }</pre>	<input type="checkbox"/>	<input type="checkbox"/>	1	F
El resultado del cálculo de la complejidad temporal en el mejor caso de un algoritmo X, da como resultado $n + n \cdot \log(n)$. Por lo tanto, diremos que la complejidad del algoritmo X cuando $n \rightarrow \infty$ pertenece a $\Omega(n)$.	<input type="checkbox"/>	<input type="checkbox"/>	2	F
Las pilas también se conocen como listas LIFO.	<input type="checkbox"/>	<input type="checkbox"/>	3	V
Dado un único recorrido de un árbol binario lleno, es posible reconstruir dicho árbol.	<input type="checkbox"/>	<input type="checkbox"/>	4	V
A los árboles generales también se les llama árboles multicamino de búsqueda.	<input type="checkbox"/>	<input type="checkbox"/>	5	F
Cuando se realiza una inserción en un AVL, en el camino de vuelta atrás para actualizar los factores de equilibrio, como mucho solo se va a efectuar una rotación.	<input type="checkbox"/>	<input type="checkbox"/>	6	V
La altura de un árbol 2-3 únicamente crece cuando se inserta un elemento y todos los nodos del árbol son 3-nodo.	<input type="checkbox"/>	<input type="checkbox"/>	7	F
Con las operaciones de inserción y borrado es posible conseguir un árbol 2-3-4 de altura 4 con todos sus nodos de tipo 2-nodo.	<input type="checkbox"/>	<input type="checkbox"/>	8	F
Las operaciones de transformación cuando se inserta un elemento en un árbol 2-3-4, en el caso de un árbol rojo-negro, se reducen a cambios de colores o rotaciones.	<input type="checkbox"/>	<input type="checkbox"/>	9	V
El árbol 2-3 es un árbol B m-camino de búsqueda con $m=2$.	<input type="checkbox"/>	<input type="checkbox"/>	10	F
La dispersión abierta elimina el problema del clustering secundario.	<input type="checkbox"/>	<input type="checkbox"/>	11	V
Sea una tabla de dispersión cerrada con estrategia de redistribución $h_i(x) = (H(x) + C \cdot i) \text{ MOD } B$, con $B=1000$ y $C=74$. Para cualquier clave "x" se recorrerán todas las posiciones de la tabla buscando una posición libre cuando se inserta el elemento.	<input type="checkbox"/>	<input type="checkbox"/>	12	F
El siguiente árbol es un montículo máximo: <pre>graph TD 10((10)) --- 7((7)) 10 --- 9((9)) 10 --- 6((6)) 7 --- 5((5)) 7 --- 2((2)) 9 --- 3((3)) 9 --- 8((8)) 8 --- 4((4)) 4 --- 1((1))</pre>	<input type="checkbox"/>	<input type="checkbox"/>	13	F
Para todo nodo de un árbol Leftist, se cumple que el número de nodos de su hijo izquierdo es mayor o igual que el de su hijo derecho.	<input type="checkbox"/>	<input type="checkbox"/>	14	F
Un grafo no dirigido de n vértices es un árbol si está libre de ciclos y tiene $n-1$ aristas.	<input type="checkbox"/>	<input type="checkbox"/>	15	V

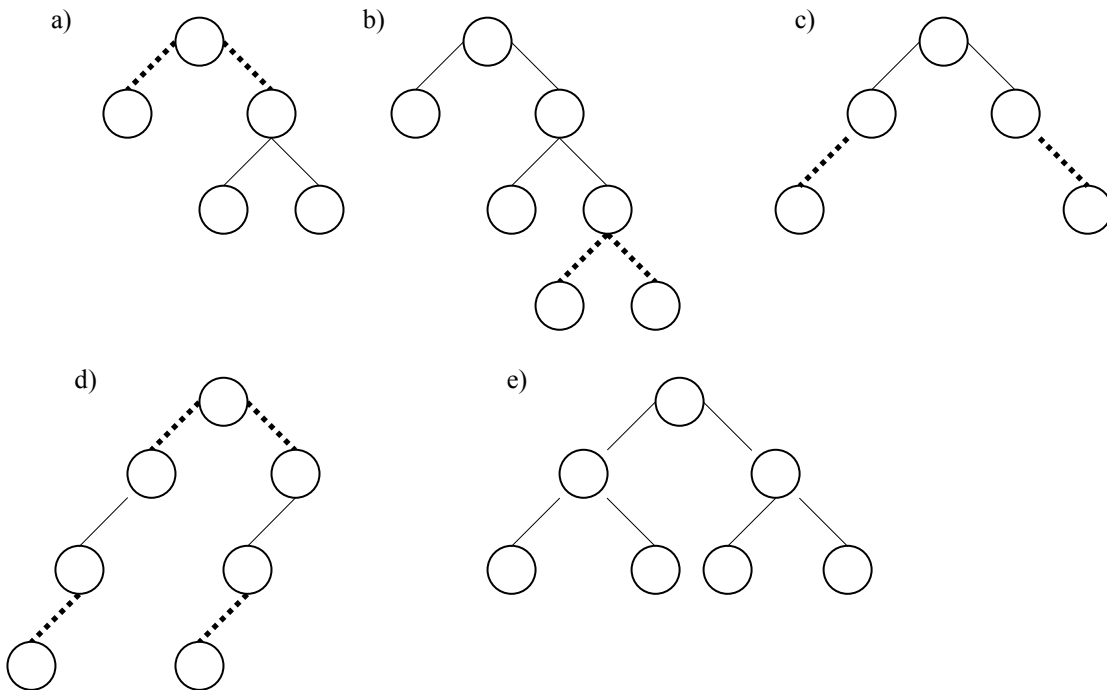
Examen PED diciembre 2009

- 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:** el jueves 5 de noviembre. Revisión de exámenes el martes 10 de noviembre de 9:30 a 10:30 en la sala de reuniones 'Claude Shannon' (sótano de la EPS IV). Examen de prácticas el jueves 12 de noviembre de 15:00 a 17:00 en los laboratorios L25 y L27 de la EPS I.
- 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 cola, escribe la sintaxis y semántica de la operación $M()$ que recibe dos colas y devuelve una cola nueva en la que se han encolado de forma alternada los elementos de las dos colas, empezando por la primera cola. Por ejemplo:

$$C1 = (a, b, c, d) \quad C2 = (1, 2, 3) \\ M(C1, C2) = (a, 1, b, 2, c, 3, d)$$

2. De los siguientes árboles determinar cuáles son R-N razonando la respuesta:



3. Insertar en una tabla de dispersión cerrada de tamaño $B=7$ los siguientes elementos: 10, 3, 17, 23, 21, 29 y 28. Mostrar el cálculo para insertar cada elemento y la tabla final con la inserción de todos los elementos:

- Con estrategia de redistribución aleatoria ($c=2$).
- Con estrategia de redistribución *segunda función hash*.

4. Escribe en C++ la forma canónica de las clases de un trie (junto con las clases de los objetos que éste contenga) que tiene las siguientes complejidades para la operación de búsqueda de una palabra: $\Omega(1)$ y $O(n * L)$, siendo n el número de letras diferentes que pueden formar parte de una palabra (por ejemplo las letras del alfabeto y los números), y L la longitud máxima de una palabra (por ejemplo de la palabra "maxilofacial", $L=12$).

NOTA:

- Hay que explicar en forma de comentarios cada campo privado de la clase.
- En la representación propuesta, explicar cuándo se producen los casos de complejidad mínima y máxima.
- El código C++ que se presente ha de ser directamente compilable, es decir, los errores de sintaxis de C++ se puntuarán negativamente.

Examen PED diciembre 2009. Soluciones

1.

**** Sintaxis:**

$M(\text{cola}, \text{cola}) \rightarrow \text{cola}$

$\text{Aux1}(\text{cola}, \text{cola}, \text{cola}) \rightarrow \text{cola}$

$\text{Aux2}(\text{cola}, \text{cola}, \text{cola}) \rightarrow \text{cola}$

**** Semántica:**

Var c, c1, c2: cola; x: ítem;

$M(c1, c2) = \text{Aux1}(\text{crear}(), c1, c2)$

$\text{Aux1}(c, \text{crear}(), \text{crear}()) = c$

$\text{Aux1}(c, \text{encolar}(c1, x), \text{crear}()) = \text{Aux1}(\text{encolar}(c, \text{cabeza}(c1)), \text{desencolar}(\text{encolar}(c1, x)), \text{crear}())$

$\text{Aux1}(c, \text{crear}(), \text{encolar}(c1, x)) = \text{Aux1}(\text{encolar}(c, \text{cabeza}(c1)), \text{crear}(), \text{desencolar}(\text{encolar}(c1, x)))$

$\text{Aux1}(c, \text{encolar}(c1, x), c2) = \text{Aux2}(\text{encolar}(c, \text{cabeza}(c1)), \text{desencolar}(\text{encolar}(c1, x)), c2)$

$\text{Aux2}(c, \text{crear}(), \text{crear}()) = c$

$\text{Aux2}(c, \text{encolar}(c1, x), \text{crear}()) = \text{Aux2}(\text{encolar}(c, \text{cabeza}(c1)), \text{desencolar}(\text{encolar}(c1, x)), \text{crear}())$

$\text{Aux2}(c, \text{crear}(), \text{encolar}(c1, x)) = \text{Aux2}(\text{encolar}(c, \text{cabeza}(c1)), \text{crear}(), \text{desencolar}(\text{encolar}(c1, x)))$

$\text{Aux2}(c, c1, \text{encolar}(c2, x)) = \text{Aux1}(\text{encolar}(c, \text{cabeza}(c2)), c1, \text{desencolar}(\text{encolar}(c2, x)))$

Solución (otra forma):

**** Sintaxis:**

$M(\text{cola}, \text{cola}) \rightarrow \text{cola}$

$\text{Concatena}(\text{cola}, \text{cola}) \rightarrow \text{cola}$

**** Semántica:**

Var c1, c2: cola;

$M(\text{crear}(), c1) = c1$

$M(c1, \text{crear}()) = c1$

$M(c1, c2) = \text{Concatena}(\text{encolar}(\text{encolar}(\text{crear}(), \text{cabeza}(c1)), \text{cabeza}(c2)), M(\text{desencolar}(c1), \text{desencolar}(c2)))$

$\text{Concatena}(\text{crear}(), c1) = c1$

$\text{Concatena}(c1, \text{crear}()) = c1$

$\text{Concatena}(c1, c2) = \text{concatena}(\text{encolar}(c1, \text{cabeza}(c2)), \text{desencolar}(c2))$

2.

a) No es R-N porque no todos los caminos desde la raíz a las hojas tienen el mismo número de nodos negros

b) Mismo razonamiento que a)

c) Sí es R-N, todos los caminos desde la raíz tienen el mismo número de nodos negros

d) Sí es R-N, mismo razonamiento que b)

e) Sí es R-N, todos los nodos del 2-3-4 equivalente son de tipo 2-nodo

3.

a) $h_i(x) = (H(x) + c \cdot i) \text{ MOD } B = (h_{i-1}(x) + c) \text{ MOD } B$

$H(10) = x \text{ MOD } B = 10 \text{ MOD } 7 = 3$

$H(3) = 3 \text{ MOD } 7 = 3$

$h_1(3) = (3+2) \text{ MOD } 7 = 5$

$H(17) = 17 \text{ MOD } 7 = 3$

$h_1(17) = (3+2) \text{ MOD } 7 = 5$

$h_2(17) = (5+2) \text{ MOD } 7 = 0$

$H(23) = 23 \text{ MOD } 7 = 2$

$H(21) = 21 \text{ MOD } 7 = 0$

$h_1(21) = (0+2) \text{ MOD } 7 = 2$

$h_2(21) = (2+2) \text{ MOD } 7 = 4$

$H(29) = 29 \text{ MOD } 7 = 1$

$H(28) = 28 \text{ MOD } 7 = 0$

$h_1(28) = (0+2) \text{ MOD } 7 = 2$

$h_2(28) = (2+2) \text{ MOD } 7 = 4$

$h_3(28) = (4+2) \text{ MOD } 7 = 6$

(0) 17 (1) 29 (2) 23 (3) 10 (4) 21 (5) 3 (6) 28

b) $k(x) = (x \text{ MOD } (B-1)) + 1 // h_i(x) = (H(x) + k(x) \cdot i) \text{ MOD } B = (h_{i-1}(x) + k(x)) \text{ MOD } B$

$H(10) = x \text{ MOD } B = 10 \text{ MOD } 7 = 3$

$$\begin{aligned}
H(3) &= 3 \bmod 7 = 3 \\
K(3) &= (3 \bmod 6) + 1 = 4 \\
h_1(3) &= (3+4) \bmod 7 = \mathbf{0} \\
H(17) &= 17 \bmod 7 = 3 \\
K(17) &= (17 \bmod 6) + 1 = 6 \\
h_1(17) &= (3+6) \bmod 7 = \mathbf{2} \\
H(23) &= 23 \bmod 7 = 2 \\
K(23) &= (23 \bmod 6) + 1 = 6 \\
h_1(23) &= (2+6) \bmod 7 = \mathbf{1} \\
H(21) &= 21 \bmod 7 = 0 \\
K(21) &= (21 \bmod 6) + 1 = 4 \\
h_1(21) &= (0+4) \bmod 7 = \mathbf{4} \\
H(29) &= 29 \bmod 7 = 1 \\
K(29) &= (29 \bmod 6) + 1 = 6 \\
h_1(29) &= (1+6) \bmod 7 = 0 \\
h_2(29) &= (0+6) \bmod 7 = \mathbf{6} \\
H(28) &= 28 \bmod 7 = 0 \\
K(28) &= (28 \bmod 6) + 1 = 5 \\
h_1(28) &= (0+5) \bmod 7 = \mathbf{5} \\
(0) \ 3 \ (1) \ 23 \ (2) \ 17 \ (3) \ 10 \ (4) \ 21 \ (5) \ 28 \ (6) \ 29
\end{aligned}$$

4.

Caso $\Omega(1)$: al buscar la palabra “zoo” si en el trie solo hay almacenada esa palabra que empiece por “z”

Caso $O(n*L)$: caso en que se busque una palabra de longitud L y en cada nodo haya que recorrer los n nodos de cada lista de strings.

```
class TNodeTrie;
```

```
class TListNode {
```

```
public:
```

```
    // Constructor por defecto
```

```
    TListNode();
```

```
    // Constructor de copia
```

```
    TListNode(const TListNode &);
```

```
    // Destructor
```

```
    ~TListNode();
```

```
    // Sobrecarga del operador asignación
```

```
    TListNode & operator=(const TListNode &);
```

```
private:
```

```
    // Letra o string a la que se corresponde el nodo
```

```
    char* c;
```

```
    // Será cierto si el camino desde la raíz hasta este nodo contiene una palabra almacenada en el trie
```

```
    bool pal;
```

```
    // Puntero al siguiente nivel del trie
```

```
    TNodeTrie* sigLetra;
```

```
    // Puntero al siguiente nodo de la lista
```

```
    TListNode* sig;
```

```
};
```

```
class TLista {
```

```
public:
```

```
    // Constructor por defecto
```

```
    TLista();
```

```
    // Constructor de copia
```

```
    TLista(const TLista &);
```

```

        // Destructor
        ~TLista();

        // Sobrecarga del operador asignación
        TLista & operator=(const TLista &);
private:
        // Puntero al primer nodo de la lista
        TListaNodo* primero;
};

class TNodeTrie {
public:
        // Constructor por defecto
        TNodeTrie();

        // Constructor de copia
        TNodeTrie(const TNodeTrie &);

        // Destructor
        ~TNodeTrie();

        // Sobrecarga del operador asignación
        TNodeTrie & operator=(const TNodeTrie &);
private:
        // Lista con cada uno de las posibles letras de una palabra
        TLista listaCaracteres;
};

class TTrie {
public:
        // Constructor por defecto
        TTrie();

        // Constructor de copia
        TTrie(const TTrie &);

        // Destructor
        ~TTrie();

        // Sobrecarga del operador asignación
        TTrie & operator=(const TTrie &);
private:
        // Puntero a la raíz del trie
        TNodeTrie* raiz;
};

```